Lecture 18:
More NP-Complete Problems
Polynomial Time Reducibility

$f : \Sigma^* \rightarrow \Sigma^*$ is a polynomial time computable function
if there is a poly-time Turing machine $M$ that on every input $w$, halts with just $f(w)$ on its tape

Language $A$ is poly-time reducible to language $B$, written as $A \leq_p B$, if there is a poly-time computable $f : \Sigma^* \rightarrow \Sigma^*$ so that:

$$w \in A \iff f(w) \in B$$

$f$ is a polynomial time reduction from $A$ to $B$

Note there is a $k$ such that for all $w$, $|f(w)| \leq k|w|^k$
**Definition:** A language $B$ is **NP-complete** if:

1. $B \in \text{NP}$

2. Every $A$ in NP is poly-time reducible to $B$
   That is, $A \leq_p B$
   When this is true, we say “$B$ is NP-hard”

Today we’ll see many more NP-complete problems:
- NHALT, 3SAT, **CLIQUE**, IS, VC, SUBSET-SUM,
- KNAPSACK, PARTITION, BIN-PACKING, ... 
(6.890: an entire class on this kind of stuff)

And even more on pset/pests...
Given a graph G and positive k, does G contain a complete subgraph on k nodes?

CLIQUE = \{ (G,k) \mid G \text{ is an undirected graph with a k-clique} \}
The Clique Problem

Given a graph G and positive k, does G contain a complete subgraph on k nodes?

CLIQUE = { (G,k) | G is an undirected graph with a k-clique }

Theorem (Karp): CLIQUE is NP-complete

Why is it in NP?
Theorem: CLIQUE is NP-Complete
\[ 3\text{SAT} \leq_p \text{CLIQUE} \]

Transform every 3-cnf formula \( \phi \) into \((G,k)\) such that

\[ \phi \in 3\text{SAT} \iff (G,k) \in \text{CLIQUE} \]

Want transformation that can be done in time that is polynomial in the length of \( \phi \)

How can we encode a logic problem as a graph problem?
3SAT $\leq_P$ CLIQUE

We transform any 3-cnf formula $\phi$ into $(G,k)$ such that

$$\phi \in 3SAT \iff (G,k) \in CLIQUE$$

Let $C_1, C_2, \ldots, C_m$ be clauses of $\phi$, let $x_1, \ldots, x_n$ be vars. Set $k := m$

Make a graph $G$ with $m$ groups of 3 nodes each.

Idea: Group $i$ corresponds to clause $C_i$ of $\phi$

Each node in group $i$ is “labeled” by a literal of $C_i$

(Note these labels do not actually appear in the graph!)

Put edges between all pairs of nodes in different groups, except for pairs of nodes with labels $x_i$ and $\neg x_i$

Put no edges between nodes in the same group
\[(x_1 \lor x_1 \lor x_2) \land (\neg x_1 \lor \neg x_2 \lor \neg x_2) \land (\neg x_1 \lor x_2 \lor x_2)\]

\[|V| = 3\text{(number of clauses)} \quad k = \text{number of clauses}\]
\[(x_1 \lor x_1 \lor x_1) \land (\neg x_1 \lor \neg x_1 \lor x_2) \land (x_2 \lor x_2 \lor x_2) \land (\neg x_2 \lor \neg x_2 \lor x_1)\]
Claim: $\phi \in 3\text{SAT} \iff (G,m) \in \text{CLIQUE}$

Claim: If $\phi \in 3\text{SAT}$ then $(G,m) \in \text{CLIQUE}$

Proof: Let $A$ be a SAT assignment of $\phi$.
For each clause $C$ of $\phi$, some literal in $C$ is set true by $A$.
For each clause $C$, let $v_C$ be one vertex from the group for $C$ in $G$, where the label of $v_C$ is a literal set true by $A$.

Claim: $S = \{v_C \mid C \in \phi\}$ is an $m$-clique in $G$. (note $|S| = m$)

Proof: Let $v_C \neq v_{C'}$ be in $S$. Suppose $(v_C, v_{C'}) \notin E$.
Then $v_C$ and $v_{C'}$ are from different groups, so they must label \textit{inconsistent} literals, call these literals $x$ and $\neg x$.
But assignment $A$ cannot set true both $x$ and $\neg x$! Contradiction. So $(v_C, v_{C'}) \in E$, for all $v_C, v_{C'} \in S$.
Hence $S$ is an $m$-clique, and $(G,m) \in \text{CLIQUE}$.
Claim: \( \phi \in 3\text{SAT} \iff (G,m) \in \text{CLIQUE} \)

Claim: If \((G,m) \in \text{CLIQUE}\) then \(\phi \in 3\text{SAT}\)

Proof: Let \(S\) be an \(m\)-clique of \(G\).

We’ll construct a satisfying assignment \(A\) of \(\phi\).

Claim: \(S\) contains exactly one node from each group of \(G\).

For each variable \(x\) of \(\phi\), define variable assignment \(A\): \(A(x) := 1\), if there is a vertex in \(S\) with label \(x\), \(A(x) := 0\), if there is a vertex in \(S\) with label \(\neg x\), or no vertices in \(S\) are labeled \(x\) or \(\neg x\)

For all \(i = 1,\ldots,m\), one vertex from the \(i\)-th group is in \(S\).

\(\Rightarrow\) one literal from the \(i\)-th clause of \(\phi\) is a vertex in \(S\)

So for all \(i = 1,\ldots,m\), \(A\) sets at least one literal true in \(i\)-th clause of \(\phi\). Therefore \(A\) is a satisfying assignment to \(\phi\).
Independent Set is NP-hard

**IS:** Given a graph \( G = (V, E) \) and integer \( k \), is there \( S \subseteq V \) such that \( |S| \geq k \) and no pair of vertices in \( S \) have an edge?

**CLIQUE:** Given \( G = (V, E) \) and integer \( k \), is there \( S \subseteq V \) such that \( |S| \geq k \) and every pair of vertices in \( S \) have an edge?

**CLIQUE \( \leq_p IS: \)**
Given \( G = (V, E) \), output \( G' = (V, E') \) where
\[
E' = \{(u,v) \mid (u,v) \notin E}\.
\]

\( (G, k) \in CLIQUE \iff (G', k) \in IS \)
a k-Clique in \( G \) = a k-IS in \( G' \)
The Vertex Cover Problem

vertex cover = set of nodes C that cover all edges
For all edges, at least one endpoint is in C
VERTEX-COVER = \{ (G,k) \mid G \text{ is a graph with a vertex cover of size at most } k \}\}

**Theorem:** VERTEX-COVER is NP-Complete

1. VERTEX-COVER ∈ NP
2. IS ≤_P VERTEX-COVER

Want to transform a graph G and integer k into G’ and k’ such that

\((G,k) \in \text{IS} \iff (G’,k’) \in \text{VERTEX-COVER}\)
Claim: For every graph $G = (V, E)$, and subset $S \subseteq V$, $S$ is an independent set if and only if $(V - S)$ is a vertex cover.

Proof: $S$ is an independent set

\[ \iff (\forall u, v \in V)[ (u \in S \text{ and } v \in S) \Rightarrow (u,v) \notin E ] \]
\[ \iff (\forall u, v \in V)[ (u,v) \in E \Rightarrow (u \notin S \text{ or } v \notin S) ] \]
\[ \iff (V - S) \text{ is a vertex cover!} \]

Therefore $(G,k) \in IS \iff (G, |V| - k) \in VERTEX-COVER$

Our polynomial time reduction: $f(G,k) := (G, |V| - k)$
The Subset Sum Problem

Given: Set $S = \{a_1, ..., a_n\}$ of positive integers and a positive integer $t$

Is there an $A \subseteq \{1, ..., n\}$ such that $t = \sum_{i \in A} a_i$?

$\text{SUBSET-SUM} = \{(S, t) \mid \exists S' \subseteq S \text{ s.t. } t = \sum_{b \in S'} b\}$

A simple summation problem!

Theorem (in algs): There is a $O(n \cdot t)$ time algorithm for solving SUBSET-SUM.

But $t$ can be specified in $(\log t)$ bits... this isn’t an algorithm that runs in polytime in the input!
The Subset Sum Problem

Given: Set $S = \{a_1, ..., a_n\}$ of positive integers and a positive integer $t$

Is there an $A \subseteq \{1, ..., n\}$ such that $t = \sum_{i \in A} a_i$?

$\text{SUBSET-SUM} = \{(S, t) \mid \exists S' \subseteq S \text{ s.t. } t = \sum_{b \in S'} b\}$

A simple summation problem!

Theorem: SUBSET-SUM is NP-complete
VC $\leq_p$ SUBSET-SUM

Want to reduce a graph to a set of numbers

Given $(G, k)$, let $E = \{e_0, \ldots, e_{m-1}\}$ and $V = \{1, \ldots, n\}$

Our subset sum instance $(S, t)$ will have $|S| = n+m$

“Edge numbers”:
For every $e_j \in E$, put $b_j = 4^j$ in $S$

“Node numbers”:
For every $i \in V$, put $a_i = 4^m + \sum_{j \in e_j} 4^j$ in $S$

Set the target number: $t = k \cdot 4^m + \sum_{j=0}^{m-1} (2 \cdot 4^j)$

Think of the numbers as being in “base 4”... as vectors with $m+1$ components
Claim: If \((G,k) \in \text{VC}\) then \((S,t) \in \text{SUBSET-SUM}\)

Suppose \(C \subseteq V\) is a VC with \(k\) vertices.

Let \(S' = \{a_i : i \in C\} \cup \{b_j : |e_j \cap C| = 1\}\)

\(S' = (\text{node numbers corresponding to nodes in C}) \text{ plus}\)
\(\text{(edge numbers corresponding to edges covered only once by C)}\)

Claim: The sum of all numbers in \(S'\) equals \(t\)

\[
\sum_{i \in C} a_i = k \cdot 4^m + \sum_{i \in C} (\sum_{j : i \in e_j} 4^j) \\
= k \cdot 4^m + \sum_{j : e_j \text{ covered once by } C} 4^j + \sum_{j : e_j \text{ covered twice by } C} (2 \cdot 4^j)
\]

\[
\sum_{j : |e_j \cap C| = 1} b_j = \sum_{j : e_j \text{ covered once by } C} 4^j \quad \text{Total sum is } t
\]
For every $e_j \in E$ ($j=0,\ldots,m-1$) put $b_j = 4^j$ in $S$

For every $i \in V$, put $a_i = 4^m + \sum_{j : i \in e_j} 4^j$ in $S$

Set $t = k \cdot 4^m + \sum_{j=0}^{m-1} (2 \cdot 4^j)$

Claim: If $(S,t) \in \text{SUBSET-SUM}$ then $(G,k) \in \text{VC}$

Suppose $C \subseteq V$ and $F \subseteq E$ satisfy

$$\sum_{i \in C} a_i + \sum_{e_j \in F} b_j = t = k \cdot 4^m + \sum_{j=0}^{m-1} (2 \cdot 4^j)$$

Claim: $C$ is a vertex cover of size $k$.

Proof: Subtract the $b_j$ numbers from the LHS.

Each $b_j = 4^j$. So what remains is a sum of the form:

$$\sum_{i \in C} a_i = k \cdot 4^m + \sum_{j=0}^{m-1} (c_j \cdot 4^j)$$

where each $c_j > 0$. But $c_j =$ number of nodes in $C$ covering $e_j$

Therefore every $e_j$ is covered by $C$, so $C$ is a vertex cover!

Moreover, $|C| = k$: each $a_i$ in $C$ adds $4^m$ to $t$
The Knapsack Problem

Given: \( S = \{(v_1, c_1), \ldots, (v_n, c_n)\} \) of pairs of positive integers (items)
a capacity budget \( C \)
a value target \( V \)

Is there an \( S' \subseteq \{1, \ldots, n\} \) such that
\[
(\sum_{i \in S'} v_i) \geq V \text{ and } (\sum_{i \in S'} c_i) \leq C
\]?

Define: \( \text{KNAPSACK} = \{(S, C, V) \mid \text{the answer is yes}\} \)

A classic economics/logistics/OR problem!

Theorem: \( \text{KNAPSACK} \) is NP-complete
KNAPSACK is NP-complete

KNAPSACK is in NP?

Theorem: SUBSET-SUM \( \leq_p \) KNAPSACK

Proof: Given an instance \((S = \{a_1, \ldots, a_n\}, t)\) of SUBSET-SUM, create a KNAPSACK instance:

For all \(i\), set \((p_i, c_i) := (a_i, a_i)\)

Define \(T = \{(p_1, c_1), \ldots, (p_n, c_n)\}\)

Define \(C := V := t\)

Then, \((S, t) \in \) SUBSET-SUM \(\iff (T, C, P) \in \) KNAPSACK

Subset of \(S\) that sums to \(t = \)

Solution to the Knapsack instance!
The Partition Problem

Given: Set $S = \{a_1, \ldots, a_n\}$ of positive integers

Is there an $S' \subseteq S$ such that $(\sum_{a_i \in S'} a_i) = (\sum_{a_i \in S-S'} a_i)$?

(Formally, PARTITION is the set of all $S$ such that the answer to this question is yes.)

In other words, is there a way to partition $S$ into two parts so that both parts have equal sum?

A problem in fair division:
Divide up a set of items so that two parties get item sets of the same value

Theorem: PARTITION is NP-complete
PARTITION is NP-complete

(1) PARTITION is in NP

(2) SUBSET-SUM \(\leq_p\) PARTITION

Input: Set \(S = \{a_1, \ldots, a_n\}\) of positive integers
positive integer \(t\)

Output: \(T := \{a_1, \ldots, a_n, 2A-t, A+t\}\), where \(A := \sum_i a_i\)

Claim: \((S, t) \in\) SUBSET-SUM \(\iff\) \(T \in\) PARTITION

That is, \(S\) has a subset summing to \(t\)
\(\iff\) \(T\) can be partitioned into two sets of equal sums
Input: Set \( S = \{a_1, \ldots, a_n\} \) of positive integers, positive \( t \)  
Output \( T := \{a_1, \ldots, a_n, 2A-t, A+t\} \), where \( A := \sum_i a_i \)

Claim: \((S,t) \in \text{SUBSET-SUM} \iff T \in \text{PARTITION}\)

What’s the sum of all numbers in \( T \)? \( 4A \)

Therefore: \( T \in \text{PARTITION} \iff \) There is a \( T' \subseteq T \) that sums to \( 2A \).

Proof of \((S,t) \in \text{SUBSET-SUM} \Rightarrow T \in \text{PARTITION} \):

If \((S,t) \in \text{SUBSET-SUM} \), let \( S' \subseteq S \) sum to \( t \).  
Then \( S' \cup \{2A-t\} \subseteq T \) sums to \( 2A \), so \( T \in \text{PARTITION} \)
Input: Set $S = \{a_1, \ldots, a_n\}$ of positive integers, positive $t$

Output $T := \{a_1, \ldots, a_n, 2A-t, A+t\}$, where $A := \sum_i a_i$

Remember: sum of all numbers in $T$ is $4A$.

**Claim:** $(S, t) \in\text{SUBSET-SUM} \iff T \in\text{PARTITION}$

$T \in\text{PARTITION} \iff$ There is a $T' \subseteq T$ that sums to $2A$.

Proof of: $T \in\text{PARTITION} \Rightarrow (S, t) \in\text{SUBSET-SUM}$

If $T \in\text{PARTITION}$, let $T' \subseteq T$ be a subset that sums to $2A$.

**Observation:** Exactly one of $\{2A-t, A+t\}$ can be in $T'$.

If $(2A-t) \in T'$, then $T' - \{2A-t\}$ sums to $t$. By Observation, the set $T' - \{2A-t\}$ is a subset of $S$. So $(S, t) \in\text{SUBSET-SUM}$

If $(A+t) \in T'$, then $(T - T') - \{2A-t\}$ sums to $(2A - (2A-t)) = t$

By Observation, $(T - T') - \{2A-t\}$ is a subset of $S$. Therefore $(S, t) \in\text{SUBSET-SUM}$
The Bin Packing Problem

Given: Set $S = \{a_1, \ldots, a_n\}$ of positive integers, a bin capacity $B$, and a target integer $K$.

*Can we partition $S$ into $K$ subsets such that each subset sums to at most $B$?*

Is there a way to pack the items of $S$ into $K$ bins, with each bin having capacity $B$?

Ubiquitous in shipping and optimization

**Theorem:** BIN PACKING is NP-complete
BIN PACKING is NP-complete

BIN PACKING is in NP?

Theorem: \( \text{PARTITION} \leq_p \text{BIN PACKING} \)

Proof: Given an instance \( S = \{a_1, \ldots, a_n\} \) of PARTITION, create an instance of BIN PACKING with:

\[
S = \{a_1, \ldots, a_n\} \\
B = (\sum_i a_i)/2 \\
k = 2
\]

Then, \( S \in \text{PARTITION} \iff (S,B,k) \in \text{BIN PACKING}: \)

Partition of \( S \) into two equal sums = Solution to the Bin Packing instance!